

REMARKS

Claims 1 and 3-20 were pending at the time of examination. Claim 1 has been amended. No new matter has been added. The applicants respectfully request reconsideration based on the foregoing amendments and these remarks.

Claim Rejections – 35 U.S.C. § 101

Claims 1 and 3-9 were rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. Claim 1 has been amended to recite the steps of

“creating the first instance when the first instance has not been identified in the one or more library object files; and
providing the first instance from the one or more library object files when the first instance has been identified in the one or more library object files.”

That is, the claim recites a concrete, tangible, and useful result, not only for when the first instance has not been identified, but also for when the instance has been identified. The applicants submit that claim 1 and 3-9 comply with the “practical application test” as described by the Examiner in the Office Action, and that the rejection under 35 U.S.C. § 101 for claims 1 and 3-9 be removed.

Claim Rejections – 35 U.S.C. § 103

Claims 1, 3-7 and 9-20 were rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 6,308,320 to Burch (hereinafter “Burch”) in view of U.S. Patent No. 5,408,665 to Fitzgerald (hereinafter “Fitzgerald”). Claim 8 was rejected under 35 U.S.C. § 103(a) as being unpatentable over Burch and Fitzgerald, as applied to claim 1, and in further view of U.S. Patent No. 6,041,180 to Perks et al. (hereinafter “Perks”). The applicants respectfully traverse these rejections.

The Examiner argues in section (B) on page 10 of the Office Action that

“the library archive being used by the linking process in Burch would have suggested that the repository of object files can be enhanced to become dynamic library for use the linking process as evidenced by Fitzgerald, in view of the well-known concept that library can help the linking in an immediate manner for dynamic creating of final object file. The combination as set forth in the rejection would expedite the retrieval of object files and resolving of relocation of data link time faster if those instance of object files would be used in a efficient manner as suggested by the techniques shown in Burch (not re-instantiating existing object files in persisted storage) and the techniques by Fitzgerald (not invoking object file in library that is not marked as needed).”

The applicants respectfully disagree. One major distinction between the applicants' invention and the cited art is that the method recited in claim 1 requires the identification of one or more instances. It should be noted that an instance is different from an object file, and thus that the claim does not refer to instances of object files, as the Examiner appears to think. An instance is a portion of source code that is specialized for a particular use or computing environment (*see* specification, page 2, lines 5-20). The instances are located in one or more library object files. That is, the instances are contained within the one or more object files. See for example page 9, lines 7-10, of the applicants' specification, which recites that "In one embodiment, the instance name extractor 202 accesses the object file 214, to extract instance names that may be available and stores these instance names in the instance name storage 206." As can be seen in the first step of claim 1, this access is accomplished by using linker symbol names for the one or more instances. As a result, in the applicant's invention, the library object files are directly accessible by the compiler, without requiring a significant amount of preparatory work, e.g., preparation of options files (*see* specification, page 8, lines 8-10). Another benefit resulting from the use of linker symbol names is that it is no longer necessary to transform between linker symbol names and programming language symbol names in order to determine whether an instance already exists in a library object file (*see* specification page 8, lines 13-16).

The applicants respectfully submit that the differences between instances and object files form a clear distinction between the invention and the cited art. All the steps recited in claim 1 are based on operations that involve instances in one way or another, where as neither Burch nor Fitzgerald discuss instances. Both Burch and Fitzgerald, as well as the Examiner's arguments recited above, are focused on linking of object files and on whether the applicants' invention would be obvious or not in view of the teachings in Burch and Fitzgerald.

In particular, as discussed in the previous response, Burch's "reuse depository" is a directory that contains a collection of compiled object files that will be reused. Nowhere in Burch is it mentioned that the individual object files in this collection contain instances available for use, as required by claim 1. Burch never mentions any of the specific steps that are recited in claim 1. Burch also suffers from many of the drawbacks that are avoided with the applicants' invention. For example, the object files (not instances) in Burch's repository are generated from "intermediate files 122," which are in turn generated from "source code 118" by "source code compilers 107" (Burch, FIG. 3A). Burch's method for identifying object files (col. 10, lines 46-58) is very complex and different from the method recited in claim 1. Even if it were possible to

identify instances in Burch's method, no timesavings of the type that is accomplished in the applicants' invention would be possible, due to this complex method.

Fitzgerald was cited to cure the deficiency that "Burch does not specify that the depository to create a first instance of object file is library of object files." It should be noted that the applicants' invention does not concern instances of object files, but instances that are stored within object files. Thus, it is clear that Fitzgerald does not cure the deficiencies of Burch. Furthermore, there can be no reasonable expectation of success when combining Burch and Fitzgerald, which is required to be shown by the Examiner in order to establish a *prima facie* case of obviousness, since neither of the documents discuss the notion of identifying and using instances that are stored within object files. For at least these reasons, the rejection of claim 1 is unsupported by the art and should be withdrawn.

Claims 3-9 are all dependent from claim 1, and the rejection of these claims is unsupported by the cited art for at least the reasons discussed above with regards to claim 1, and should be withdrawn.

In rejecting claim 8, the Examiner combines Burch and Fitzgerald with Perks to establish a *prima facie* case of obviousness. However, Perks does not add anything that cures the deficiencies discussed above, and it is respectfully submitted that the rejection of claim 8 in particular be withdrawn.

Claim 10, as amended, recites "a library object file **including at least one instance available for use by the source program**, the at least one instance being **identifiable by a linker symbol name;**" and "an enhanced compiler suitable for compilation of source code, wherein the enhanced compiler accesses the library object file to identify the one instance available in the library object file." Both of these limitations recite instances and linker symbol names, which are neither shown in Burch, as discussed above with regards to claim 1. As essentially the same rationale was used by the Examiner in rejecting claim 10 as in rejecting claim 1, the discussion above with regards to claim 1 applies also to claim 10, and the rejection should be removed for at least the same reasons.

Claims 11-13 are all dependent directly or indirectly from claim 10, and the rejection of these claims is unsupported by the cited art for at least the reasons discussed above with regards to claim 10, and should be withdrawn.

Claim 14 recites "instances" and "linker symbol names" in several places, for example, the preamble states that claim 14 is "a method of compilation of source program using one or more associated library object files with instances that are available for use by the source

program." The Examiner rejected claim 14 using essentially the same rationale that was used in rejecting claim 1. Thus for at least these reasons presented above with respect to claim 1, the rejection of claim 14 is unsupported by the art and should be withdrawn.

Claims 15-16 both depend from claim 14, and the rejection of these claims is unsupported by the cited art for at least the reasons discussed above with regards to claim 14, and should be withdrawn.

Claim 17 is a *Beauregard* claim corresponding to claim 1. For reasons substantially similar to those set forth above with regards to claim 1, the applicant respectfully contends that the rejection of claim 17 is unsupported by the cited art and should be withdrawn.


Claims 18-20 all depend from claim 17, and are *Beauregard* claims corresponding to claims 3, 4, and 6, respectively. The rejection of these claims is unsupported by the cited art for at least the reasons discussed above with regards to claim 17, and should be withdrawn.

CONCLUSION

The applicants believe that all pending claims are allowable and respectfully request a Notice of Allowance for this application from the Examiner. Should the Examiner believe that a telephone conference would expedite the prosecution of this application, the undersigned can be reached at the telephone number set out below.

Respectfully submitted,

BEYER WEAVER & THOMAS, LLP


Michael J. Ferrazano
Reg. No. 44,105

P.O. Box 70250
Oakland, CA 94612-0250
(650) 961-8300